Implementations of R-NET and Character-level Embeddings on SQuAD

Stanford CS224N Default Project on IID SQuAD

Luan Jiang
luanj@stanford.edu

Abstract

Question answering (QA) has become a popular discipline within the field of natural language processing in recent years. The Stanford Question Answering Dataset (SQuAD), which consists of crowd-sourced questions on Wikipedia articles, has inspired many advancements in QA systems to correctly identify answers by selecting relevant spans of texts. In this paper, I explored two ways to improve the performance of the baseline Bidirectional Attention Flow (BiDAF) model on SQuAD: incorporate character-level embeddings with the existing word-level embeddings, and implement R-NET in place of BiDAF. Experiments have shown that character-level emebeddings enriches the understanding of components of words and provides improvement on key evaluation metrics. The implementation of R-NET also provides additional lift in model performance on SQuAD.

- CS224N Staff Mentor: Davide Giovanardi

1 Introduction

Question answering has become one of the most popular topics over recent years and its applications can be seen in many day-to-day activities such as simple web search queries and smart home devices. Question answering problems come in various forms, ranging from reading comprehension where the goal is to answer questions over a single passage of text, to open-domain QA which aims to answer questions over a large collection of documents. This paper focuses on identifying solutions for Stanford Question Answer Dataset (SQuAD) which is the former of the two types of question answering systems.

SQuAD [1] is a curated dataset of passages and questions where the answers can be extracted by selecting the correct span of text from the passage. The specific SQuAD dataset used in this paper is SQuAD 2.0, which differs from its predecessor in that it combines the initially released SQuAD questions with additional unanswerable questions. Therefore, for an NLP model to perform well on this task, it needs to determine how to select a salient span of text from the passage to correctly supply the answer to an answerable question. It also needs to understand when a question is unanswerable with the provided passage.

This paper uses two methods to improve the baseline BiDAF model: an addition of character-level word embeddings, and the implementation of R-NET in place of BiDAF. I decided to implement character-level word embeddings in addition to the existing word embeddings from GloVe because character-level embeddings enrich understandings on components of words and provide numeric representations of words in SQuAD which are out of pre-trained GloVe vocabulary. Furthermore, the original BiDAF architecture also included character-level embeddings in its embed layer [2].

On the other hand, the choice of R-NET as the NLP model to implement for this paper may deserve elaboration. Even though there have been other model architectures such as model pre-training which gained immense popularity and significantly improved performance on SQuAD in recent years, for this paper, I have decided to focus on the fundamentals of implementing algorithms from scratch based on ideas proposed in previous papers. R-NET is attractive in this regard as it utilizes classical

concepts such as recurrent neural networks and self-attention. In addition, the official codebase for R-NET has not been released, providing opportunities for interpretations of implementation solely based on the published paper. Lastly, the published paper on R-NET achieved admirable result on SQuAD 1.1 amongst other non-pretrained NLP models. However, its performance has not been tested on the unanswerable questions in SQuAD 2.0. Therefore, evaluating R-NET on SQuAD 2.0 may offer additional insights on the model's understanding of human language.

2 Related Work

The baseline model used in this paper is an implementation of Bidirectional Attention Flow (BiDAF) [2] with word-level embeddings only. The first improvement made in my implementation is to incorporate character-level word embeddings in addition to the existing word embeddings inspired and simplified from Kim et al.'s paper in 2016 [3].

The implementation of R-NET in this paper largely follows the approach described in the original paper published on R-NET [4]. The design of R-NET drew inspiration from the attention-based recurrent network in match-LSTM [5] but adds an additional gate to better capture the relevance of passage components to the question. In addition, it introduces a new self-matching mechanism to encode broader context within the passage.

The usage of pointer network in R-NET and in this paper is inspired by Vinyals et al. [6] which predicts the start position of the answer and feeds the prediction to a GRU to point to the corresponding end position.

3 Approach

Two models are implemented in this paper:

1. Baseline BiDAF with added character-level word embeddings.
2. The implementation of R-NET in place of BiDAF.

3.1 BiDAF with added character-level embeddings

Complete code has been provided for the baseline BiDAF model with word embeddings only. Therefore, this section only focuses on my original code contribution to BiDAF model which is an addition of character-level word embeddings. First, an embedding lookup is performed using character indexes c_idxs:

$$emb\_char = CharEmbedding(c\_idxs) \tag{1}$$

Next, dropout is performed before 1D convolution is applied. Kernel size of 3 is elected here.

$$out\_conv = Conv1d(emb\_char) \tag{2}$$

Finally, ReLU and max pooling are applied to obtain an abstracted representation of the characters.

$$emb\_char_{maxpool} = MaxPool(ReLU(out\_conv)) \tag{3}$$

$emb\_char_{maxpool}$ is concatenated with word embeddings before highway network is applied.

$$emb\_word\_char = Highway[emb\_word; emb\_char] \tag{4}$$

Even though the GloVe vector representation of words used for this paper contains almost 89k words, there are still words in SQuAD that can be out-of-vocabulary or may have been misspelled. Character-level embeddings has proven to be resilient against spelling mistakes and rare words as it can still

represent words by their character-level compositions. In addition, character-level embeddings are not as computationally expensive as word embeddings due to their smaller vocabulary size. All of these attributes make character-level word embeddings a natural addition to the baseline BiDAF model.

The addition of character-level word embeddings completes the BiDAF model architecture as seen in figure 1 which was presented in its original publication [2].

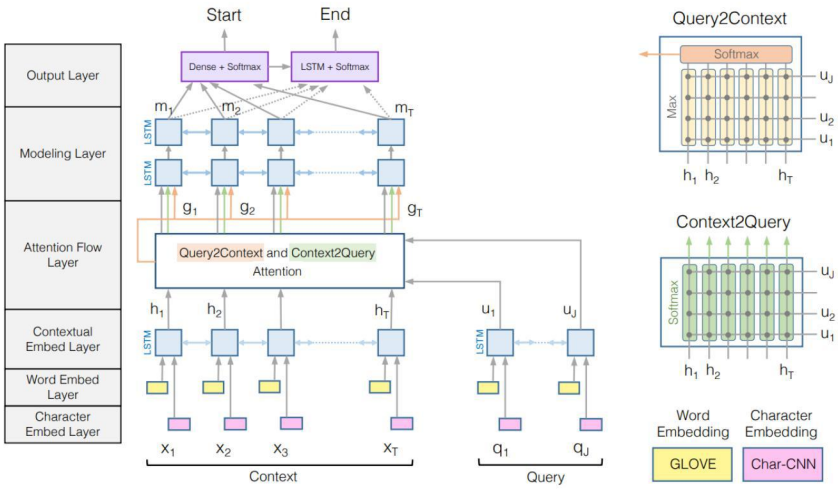

Figure 1: Model architecture of BiDAF from [2].

3.2 R-NET

The implementation of R-NET is largely based on the approach discussed in paper [4] and attempts to recreate the model architecture shown in figure 2.

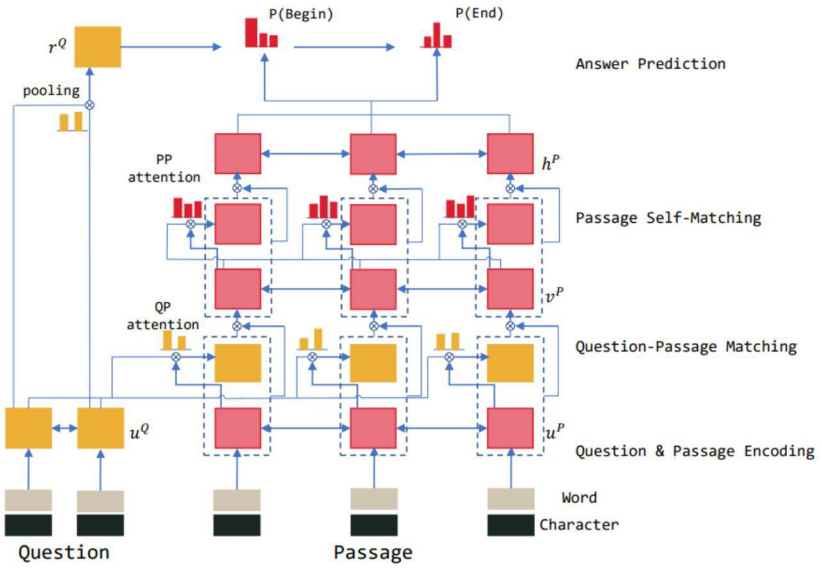

Figure 2: Model architecture of RNET from [4].

3.2.1 Question and Passage Encoding

First, a combination of word embeddings using GloVe and character-level embeddings are produced following the same approach as described in section 3.1 of this paper.

Next, the combined embeddings are passed through bi-directional RNNs to produce new representations. The same operation is performed on both passage and question embeddings.

$$u_t^Q = BiRNN_Q(u_{t-1}^Q, [embed\_word\_char_Q]) \tag{5}$$

$$u_t^P = BiRNN_P(u_{t-1}^P, [embed\_word\_char_P]) \tag{6}$$

GRU is used for computational efficiency. This completes the steps required for passage and question encoding.

3.2.2 Question-Passage Matching

The encoded question and passage representations u^Q, u^P are passed to a gated attention-based recurrent network to generate question-aware passage representations.

For each passage representation u_t^P, compute

$$\begin{aligned} s_j^t &= v^T \tanh(W^Q u^Q + W_u^P u_t^P + W_u^P v_{t-1}^P) \\ a_i^t &= Softmax(s_j^t) \\ c_t &= a_i^t * u^Q \end{aligned} \tag{7}$$

v_{t-1}^P in the above formula is the representation from the previous timestamp. Next, a gate g_t is applied based on the current passage word and its attention-pooling vector of the question. The gate effectively identifies parts of the passage that are relevant to the question.

$$\begin{aligned} g_t &= sigmoid(W_g[u_t^P, c_t]) \\ [u_t^P, c_t]^* &= g_t \odot [u_t^P, c_t] \end{aligned} \tag{8}$$

$[u_t^P, c_t]^*$ are passed to another RNN to complete the generation of question-aware passage representations. I used GRUCell in my implementation.

$$v_t^P = RNN(v_{t-1}^P, [u_t^P, c_t]^*) \tag{9}$$

3.2.3 Passage Self-Matching

A limitation with the question-aware passage representation is that it lacks awareness of the broader context of the passage outside of its immediate surrounding window which may have helped infer the answer. Therefore, R-NET introduces a passage self-matching mechanism to encode relevant information from the rest of question-aware passage representations to the current representation.

For each question-aware passage representation v_t^P,

$$\begin{aligned} s_j^t &= v^T \tanh(W^P v^P + W_v^{\hat{P}} v_t^P) \\ a_i^t &= Softmax(s_j^t) \\ c_t &= a_i^t * v^P \end{aligned} \tag{10}$$

Similar to question-passage matching, a gate is computed to control the input to the RNN.

$$\begin{aligned} g_t &= sigmoid(W_g[v_t^P, c_t]) \\ [v_t^P, c_t]^* &= g_t \odot [v_t^P, c_t] \end{aligned} \tag{11}$$

Finally, $[v_t^P, c_t]^*$ is passed to a bi-directional RNN to complete the self-matching process. I used GRUCells in my implementation of the bi-directional RNN instead of LSTMCells for computational efficiency.

$$h_t^P = BiRNN(h_{t-1}^P, [v_t^P, c_t]^*) \tag{12}$$

3.2.4 Answer Prediction with Pointer Network

First, I initialize the hidden state of the answer network with the following:

$$\begin{aligned} s &= v^T \tanh(W^Q u^Q) \\ a &= Softmax(s) \\ r^Q &= \sum_{i=1}^{m} a_i u_i^Q \end{aligned} \tag{13}$$

Next, to compute the start position of the pointers:

$$\begin{aligned} s &= w^T \tanh(W^P h^P + W^a r^Q) \\ a_{start} &= LogSoftmax(s) \end{aligned} \tag{14}$$

Using the current predicted probability of start position a_{start}, I then compute an attention-pooling layer to be used as the input to an RNN:

$$\begin{aligned} c_t &= \sum_{i=1}^{n} a_{start\_i}^t h_i^P \\ h_t^a &= RNN(h_{t-1}^a, c_t) \end{aligned} \tag{15}$$

The output hidden state h_t^a is used to compute the end position of the pointers:

$$\begin{aligned} s &= \hat{w}^T \tanh(\hat{W}^P h^P + \hat{W}^a h_t^a) \\ a_{end} &= LogSoftmax(s) \end{aligned} \tag{16}$$

4 Experiments

4.1 Data

The experiment dataset used is the Stanford Question Answering Dataset (SQuAD 2.0), which consists of questions that can be answered with a span of text from the context passage as well as questions that cannot be answered with span extraction. It contains 129,941 examples in training set, 6078 in dev set and 5915 in test set.

4.2 Evaluation method

Exact Matching (EM) and F1 score are selected as the evaluation criteria. During dev set and test set evaluation, the model-selected text is compared with three human-provided answers and the maximum EM and F1 score are used.

4.3 Experimental details

The final experiment on BiDAF model with added character-level word embeddings uses setup similar to baseline model which include Adadelta with learning rate of 0.5, batch size of 64, dropout probability of 0.2 and 30 epochs. The baseline BiDAF model required approximately 21 minutes per epoch during training on an NC6 VM. The BiDAF model with my additional implementation of

character-level embeddings doubled the training time to approximately 42 minutes per epoch during training on an NC6.

The final experiment on the R-NET model with added character-level word embeddings uses Adadelta with learning rate of 0.5, batch size of 64, dropout probability of 0.1 and just 12 epochs. The final model with full R-NET architecture caused CUDA out of memory error on NC6 but was run successfully on NC6 V2. It requires approximately 58 minutes per epoch during training on NC6 V2.

A few items to note:

1. the dropout probability of 0.1 was selected for the final experiment on the R-NET model as previous runs have shown that a dropout rate of 0.3 or greater negatively impacts model performance, while a dropout rate of 0.1 performs comparably to a dropout rate of 0.2 and trains faster.
2. The number of epochs for the final R-NET model was reduced to 12 epochs as the model started to overfit on the dev set beyond approximately 1.2M iterations (see figure 3).
3. The two bi-directional RNNs within the question and passage encoding layer was removed during one of the runs. The remaining R-NET model trained successfully on NC6 and the model performance only degraded very slightly in F1 and EM scores. This makes sense intuitively as the purpose of the bi-directional RNN in the encoding layer is to capture relevant context from the rest of the passage which is similar to the purpose of the self-matching layer. However, to be consistent with the original R-NET architecture proposed in [4], this slight redundancy is retained and the encoding layer is kept in the final model trained on NC6 V2.

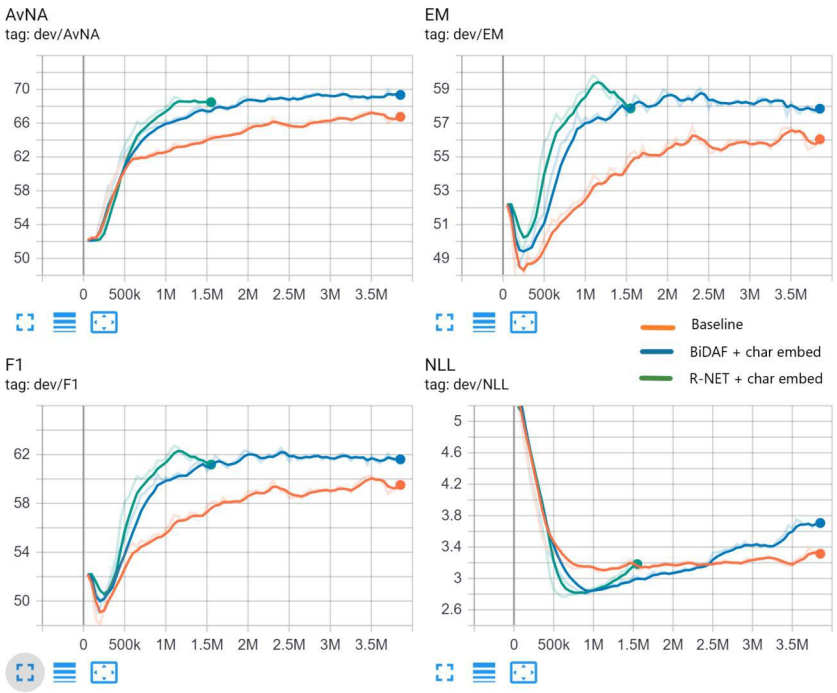

Figure 3: Tensorboard shows RNET model is overfitting after approximately 1.2M iterations

4.4 Results

Below are the baseline model scores on the dev set:

EM = 56

F1 = 58

The BiDAF model with added character-level embeddings improved on the baseline with the following result on validation leaderboard:

EM: 59.066

F1: 62.538

The R-NET model with added character-level embeddings further improved performance with the following results on the validation leaderboard:

EM: 59.822

F1: 62.702

The scores of the R-NET model with added character-level embeddings on the test leaderboard are:

EM: 58.935

F1: 62.348

In retrospect, the improvement of R-NET model compared to BiDAF, albeit small, is not surprising. Close examination revealed significant similarities in the design purposes of components of the two model architectures. The two models share similar encoding logic. The attention layer of the BiDAF model identifies relevance between query and context, achieving a similar goal as the question-passage matching layer in R-NET. The modeling layer of BiDAF captures interactions between context words, serving a similar purpose to the passage self-matching mechanism in R-NET.

5 Analysis

A shortcoming for both the BiDAF with added character embeddings and the R-NET model is that they attempt to make predictions for the questions that are unanswerable when similar phrases appear in the passage and the question. See figure 4 for an example of an incorrect prediction by the R-NET model.

- **Question:** What did Geroge Lenczowski do to the price of oil on October 16, 1973?
- **Context:** In response to American aid to Israel, on October 16, 1973, OPEC raised the posted price of oil by 70%, to $5.11 a barrel. The following day, oil ministers agreed to the embargo, a cut in production by five percent from September's output and to continue to cut production in five percent monthly increments until their economic and political objectives were met. On October 19, Nixon requested Congress to appropriate $2.2 billion in emergency aid to Israel, including $1.5 billion in outright grants. George Lenczowski notes, "Military supplies did not exhaust Nixon's eagerness to prevent Israel's collapse...This [$2.2 billion] decision triggered a collective OPEC response." Libya immediately announced it would embargo oil shipments to the United States. Saudi Arabia and the other Arab oil-producing states joined the embargo on October 20, 1973. At their Kuwait meeting, OAPEC proclaimed the embargo that curbed exports to various countries and blocked all oil deliveries to the US as a "principal hostile country".
- **Answer:** N/A
- **Prediction:** raised the posted price of oil by 70%, to $5.11 a barrel

Figure 4: An example of an incorrect prediction by the R-NET model

In this example, "Geroge Lenczowski", "the price of oil" and "October 16, 1973" appeared in both the passage and the question. The model incorrectly predicts that the answer is in the passage due to the overlapping phrases, a potential pitfall of the question-aware passage matching mechanism. The model also assumes "Geroge Lenczowski" acted on the price of oil, a potential limitation of the self-matching mechanism. One potential mitigation for this is to include additional features such as sentence structure encoding. Sentence structures may aid the model's understanding of the causal relationship between entities in the passage and arrive at the correct interpretation that "Geroge Lenczowski" did not cause the rise of oil price but merely co-occurred in the passage.

Figure 5 below shows another incorrect prediction of the R-NET model.

In this example, the model correctly extracts the information that Norman army attacked Dyrrachium, but it fails to understand the year 1185 is not in the 11th century. This mistake may be inevitable as the knowledge of centuries and their corresponding years may not have been in the SQuAD training set.

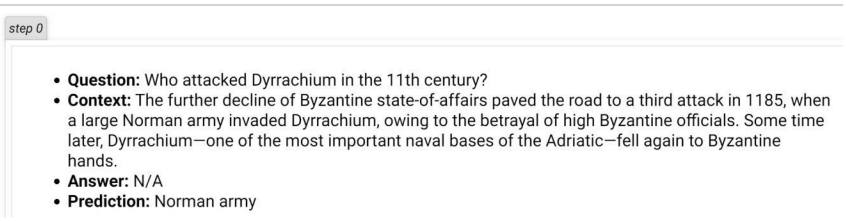

Figure 5: An example of an incorrect prediction where worldly knowledge could have helped

This is one of the instances where I expect a pre-trained model such as BERT can help as pre-training can enrich the model with additional worldly knowledge.

6 Conclusion

In this paper, I have shown that both model implementations, BiDAF with character-level embeddings and R-NET, have improved upon the baseline model performance on the SQuAD 2.0 dataset. This project has been a fruitful experience in enriching my understanding of classical deep learning concepts in the natural language domain such as recurrent neural networks and self-attention.

Nonetheless, this project also shows that while R-NET achieved admirable results on SQuAD 1.1 in its original publication [4], it struggles with correctly identifying unanswerable questions in SQuAD 2.0 when similar phrases appear in both passage and question. If given unlimited time, additional hyper-parameter tuning may have helped further improve the scores of R-NET. However, given the vast similarities between the goals and purposes of architecture layers in BiDAF and R-NET, I do think there is a ceiling on how much R-NET can outperform the baseline model. In the future, I am interested in implementing and testing some of the other models whose architectures are fundamentally different from the classical recurrent network framework, such as QANet and models with pre-trained transformers. I expect a fundamental shift in model architecture is required to truly make a huge leap on the performance on SQuAD 2.0.

References

[1] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In *Association for Computational Linguistics (ACL)*, 2018.

[2] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. In *arXiv preprint arXiv:1611.01603*, 2016.

[3] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush. Character-aware neural language models. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2016.

[4] Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. Gated self-matching networks for reading comprehension and question answering. In *Association for Computational Linguistics (ACL)*, 2017.

[5] Shuohang Wang and Jing Jiang. Learning natural language inference with lstm. In *The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016.

[6] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada, pp. 2692–2700*, 2015.